(Edited February 3, 2022)

Introduction

In this CAD you will design a 16-bit Logarithmic shifter (left shift only) for your microprocessor.

Shifter

The shifter is an essential element for many microprocessor operations. It may be used to align or scale data, manipulate bits and bytes, or it may be used in an automatic or program-controlled shift-and-add multiply function. In the baseline machine, you are only required to implement a Left Shift, which shifts data in a register (Rdest) as specified by a signed amount operand (only positive).

You must support a shift amount contained in a register or the immediate field of the instruction. This means that you will be using a mux to select the shift amount either from a register or from the immediate field. You will be using a similar mux for the ALU when you integrate the datapath.

All bits shifted out of the destination register are lost. All destination bits not mapped from the original operand are filled with zeros.

Other common shift functions, which could be added to eh baseline processor, are arithmetic shifts, arithmetic shift including the carry bit, byte swap, and rotate.

To shift the contents of a register one bit per clock cycle, an ordinary shift register, which can be assembled from cascaded D-latches, could be used. However if there is a need to shift data by an arbitrary number of bit positions within one clock cycle, a dedicated programmable shifter is required.

In this CAD, you will be designing a Logarithmic shifter.

Logarithmic Shifter

In logarithmic shifters, the total shift value is decomposed into shifts over powers of two. A shifter with a maximum shift width of M consists of $\log_2 M$ stages, where the i^{th} stage either shifts over 2^i or passes the data unchanged. Rabaey (Figure 11-38, page 597) shows a multiplexor-based logarithmic shifter. One could also use tri-state buffer multiplexors or logic gate multiplexors instead of transmission gates. A circuit like this with many transmission gates in series would benefit from having buffers inserted along the path – perhaps every three stages or so. This example is hard wired to give arithmetic right and left shifts. Minor changes will convert it to left shift only.

Logarithmic shifters have intrinsic decoding, as the shift control bits are used directly to control the muxes or transmission gates.

Procedure

Design

You need to implement a log shifter which achieves left shifts from 0 to 15 bits. Keep in mind that the shift amount must be a positive number (signed number representation). Provide buffering for any array-wide control signals.

Logic Verification

Run NCVerilog on the 16-bit shifter and verify that it functions as expected (shift left from 0 to 15 bits).

Pre-Layout Simulations

As with earlier designs, build a worst-case path schematic for device sizing purposes prior to starting layout. You may need to add some significant capacitance to emulate the wire capacitance. Sizing up device widths will help to some extent.

Layout

Make sure that the shifter is bit-slide width-matched with the rest of your datapath structures. You may find the layout of the shifter to be wire intensive, so think first of how to run metal lines.

Design Verification

Run DRC & LVS on the entire shifter. Extract parasitics.

Post-Layout Simulation

Find the delays through your shifter in simulation with parasitics. Report the worst case delay.

Submission

Please refer to the submission guidelines mentioned in the CAD1 assignment, as they apply to all CAD assignments. Do not lose points over trivial mistakes like incorrect file names, etc!

For CAD5, you need to submit the following:

- Log Shifter:
  - Schematic of the shifter, including drivers/buffers
  - Layout of the shifter. This should include hierarchical layout of the drivers/buffers.
  - Clean DRC report file.
  - Clean LVS report file.
  - PEX (Calibre view).

  - NCVerilog waveform showing a fixed input (10101010) with all possible shift amounts (single file).
  - NCVerilog waveform (the same as above but with delay added).
  - Analog simulation printouts showing critical path delay (after PEX, rise/fall)

- README:
  - Full path to your designs.
  - Briefly explain how you implemented your shifter
  - Worst case delay numbers (rise/fall) and explains the worst case path
  - Estimate all input pin capacitances, and explain how you estimated these numbers
  - Other comments
- You need to name your directory CAD5 and create a SUBMIT directory in your CAD5 directory. Copy all of your DRC, LVS reports and simulation waveform files to the SUBMIT directory.

Deadline

You must complete CAD5 by **February 17, 2021** by **11pm**.
Do not modify any files in your CAD directory after that time.